Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Υπολογιστική Κρυπτογραφία

(ΣΗΜΜΥ, ΣΕΜΦΕ, ΑΛΜΑ, ΕΜΕ)

3η Σειρά Ασκήσεων

Προθεσμία παράδοσης: 10/12/2020

Άσκηση 1. Θεωρήστε την παραλλαγή του DES-X, με 2 κλειδιά k_1, k_2, όπου η κρυπτογράφηση ενός απλού κειμένου M γίνεται ως εξής :

$$Enc_{k_1,k_2}(M) = E_{k_1}(M \oplus k_2),$$

όπου E η συνάρτηση κρυπτογράφησης του DES.

Έχουμε περισσότερη ασφάλεια από τον κλασσικό DES στο παραπάνω σύστημα; Θεωρήστε ότι ο αντίπαλος έχει δυνατότητα KPA (διαθέτει αρκετά ζεύγη απλού κειμένου - κρυπτοκειμένου).

Άσκηση 2. Κάποιος σκέφτηκε να χρησιμοποιήσει ένα ασφαλές κρυπτοσύστημα τμήματος (π.χ. AES) για ταυτόχρονη κρυπτογράφηση και έλεγχο ακεραιότητας χρησιμοποιώντας τον τρόπο λειτουργίας CBC ως εξής: ο αποστολέας προσθέτει ένα επιπλέον block, αποτελούμενο από '0' μόνο, στο τέλος του απλού κειμένου και κρυπτογραφεί με CBC mode. Αν ο παραλήπτης κατά την αποκρυπτογράφηση πάρει το ίδιο block (όλο '0') στο τέλος του αποκρυπτογραφημένου κειμένου, θεωρεί ότι το μήνυμα μεταδόθηκε σωστά. Εξασφαλίζει αυτή η μέθοδος την ακεραιότητα του μηνύματος;

Άσκηση 3. 'Εστω h συνάρτηση σύνοψης, η οποία συμπιέζει ακολουθίες μήκους $2n$ σε ακολουθίες μήκους n και έχει την ιδιότητα δυσκολίας εύρεσης συγκρούσεων (collision free). Θέλουμε να φτιάξουμε μία συνάρτηση κατακερματισμού που να συμπιέζει ακολουθίες μήκους $4n$ σε ακολουθίες μήκους n, η οποία να έχει επίσης την ιδιότητα δυσκολίας εύρεσης συγκρούσεων. Έχουμε τις εξής υποψήφιες:

1. $h_1(x_1||x_2||x_3||x_4) = h((x_1 \oplus h(x_2||x_2))||(h(x_3||x_3) \oplus x_4))$
2. $h_2(x_1||x_2||x_3||x_4) = h(h(x_1||x_2)||h(x_3||x_4))$
3. $h_3(x_1||x_2||x_3||x_4) = h(x_1||x_2) \oplus h(x_3||x_4)$
4. $h_4(x_1||x_2||x_3||x_4) = h(h(h(x_1||x_2)||x_3)||x_4)$

(Με "$\oplus$" συμβολίζουμε το XOR, με "||" την παράθεση και $|x_i| = n$.)

Για κάθε i εξετάστε αν η h_i έχει την ιδιότητα δυσκολίας εύρεσης συγκρούσεων ή όχι. Για να δείξετε ότι την έχει, δείξτε ότι αν μπορούσαμε να βρούμε συγκρούσεις για την h_i, τότε θα μπορούσαμε να βρούμε συγκρούσεις και για την h. Για να δείξετε το αντίθετο βρείτε μία ή περισσότερες συγκρούσεις για την h_i.

Άσκηση 4. Θέλουμε να κατασκευάσουμε μια συνάρτηση σύνοψης από δύο άλλες συναρτήσεις σύνοψης H_1, H_2 για τις οποίες γνωρίζουμε ότι τουλάχιστον η μία είναι ελεύθερη συγκρούσεων. Εξετάστε αν οι παρακάτω συναρτήσεις είναι ελεύθερες συγκρούσεων, και αν όχι πάντοτε, σε ποιες περιπτώσεις θα μπορούσαν να είναι και σε ποιες όχι.

1. $H_3(m) = H_1(m)||H_2(m)$.
2. $H_4(m) = H_1(H_2(m))$.

Άσκηση 5. Θεωρήστε την γεννήτρια ψευδοτυχαίων bit BBS με Blum integer $n = pq$.

(α) Να προσδιορίσετε επακριβώς την περίοδο της γεννήτριας. Εξηγήστε γιατί πρέπει να είναι μικρό το $\gcd(p-1, q-1)$.

(β) Οι "safe primes" είναι ειδικοί πρώτοι αριθμοί της μορφής $p = 2p' + 1$ όπου p' είναι επίσης πρώτος. Ονομάζουμε "SafeSafe primes" τους ειδικούς εκείνους πρώτους p για τους οποίους ισχύει ότι αν p'' είναι πρώτος με $p'' \equiv 1 \pmod 4$, τότε $2p''+1$: πρώτος και $p = 2(2p''+1)+1$. Ποια είναι η **μέγιστη** περίοδος της γεννήτριας στην περίπτωση που τόσο ο p όσο και ο q είναι "SafeSafe" πρώτοι; Να αποδείξετε τον ισχυρισμό σας.

Άσκηση 6. (προγραμματιστική συνέχεια της άσκησης 5)

Κατασκευάζοντας Blum integer $n = pq$ με "SafeSafe" primes p, q 20 δυαδικών ψηφίων ο καθένας όπως ορίζεται στο ερώτημα (β) της άσκησης 5, θα προσομοιώσουμε την γεννήτρια BBS, διαλέγοντας το s_0 ώστε αυτή να έχει **μέγιστη** περίοδο.

(α) Γράψτε ένα πρόγραμμα που να κατασκευάζει κατάλληλα την γεννήτρια (δηλαδή να βρίσκει "έξυπνα" κάποιο κατάλληλο s_0) για συγκεκριμένα p, q που εσείς θα έχετε διαλέξει σύμφωνα με τις παραπάνω συνθήκες.

(β) Επεκτείνετε το παραπάνω πρόγραμμα για να προσομοιώσετε τη γεννήτρια και επαληθεύστε πειραματικά την θεωρητικά υπολογιζόμενη περίοδό της.

Σημείωση: είναι πιθανόν η επαλήθευση να διαρκέσει μέχρι και 2 ώρες σε υπολογιστή. Προτείνεται για αυτό το λόγο να χρησιμοποιήσετε κάποιες από τις δυνατότητες της βιβλιοθήκης χειρισμού πολύ μεγάλων αριθμών GMP, που έχει μεταξύ άλλων και ένα φιλικό interface σε Python.

(γ) Θεωρώντας σημεία στο επίπεδο, στο τετράγωνο με κορυφές $(0,0), (255,255)$, το ποσοστό εξ αυτών που βρίσκονται εντός κύκλου με κέντρο το σημείο $(127.5, 127.5)$ και ακτίνα 127.5 είναι $\pi/4$. Εκτιμήστε θεωρητικά πόσα σημεία χρειάζεται να πάρετε για να πετύχετε ακρίβεια 2, 3, 4 δεκαδικών ψηφίων στο π μέσω αυτής της διαδικασίας.

Υπόδειξη: μπορείτε, αν θέλετε, να χρησιμοποιήσετε Chernoff bound.

(δ) Υλοποιήστε το εξής πείραμα ελέγχου ψευδοτυχαιότητας: θεωρώντας block των 64 bit σαν ζεύγη σημείων στο τετράγωνο χρησιμοποιώντας την διαδικασία του ερωτήματος (γ), συγκρίνετε τον πειραματικά υπολογιζόμενο αριθμό σημείων (από τα bit της γεννήτριάς σας) που χρειάζεστε για την επίτευξη ακρίβειας 2, 3, 4 δεκαδικών ψηφίων στο π με τον θεωρητικά αναμενόμενο αριθμό σημείων για την εκάστοτε ακρίβεια. Εκτός από την μη-τυχαιότητα της γεννήτριας, πού αλλού οφείλεται ενδεχόμενο σφάλμα για το οποίο δεν μπορείτε να λάβετε ακριβή προσέγγιση του π;

(ε) Συγκρίνετε με την παραλλαγή όπου το bit εξόδου της γεννήτριας είναι η ισοτιμία των δυαδικών ψηφίων των αριθμών x_i ($= x_{i-1}^2 \bmod n$) αντί για το λιγότερο σημαντικό τους ψηφίο.

☠ (στ) Ελέγξτε την γεννήτρια (και στις δύο εκδοχές της) με κάποια από τα τεστ που αναφέρονται σε αυτή την δημοσίευση:
`https://pdfs.semanticscholar.org/cf0a/fe558c2638db24a0527a4725fe0ae82cc88b.pdf`
Τι παρατηρείτε;

Άσκηση 7.

(α) Σας δίνεται ένα RSA κρυπτοσύστημα με τα ακόλουθα χαρακτηριστικά σε δεκαεξαδική αναπαράσταση:

```
n = 0xb844986fc061a2c0baf528a960e208832625f725fa09bfe1ac4c15bccad6031d09f8f37bf
00520bb59480070e59441ed34b7e3d118db67a035ac4b46a055a4963df4af0baa4dfab3f98566f2
c09f7c83ffec458b63931ce311241c98614659172cfe9f21ecc7d7241aea1ae1e88f796568f49a6
45ffce12c87629e8783462e5dbeb52a85c95
e = 0x369d89b820f2450462f21b02d91bcec9de528805bb22123d843fcd776ad57025980f1c335
9d45d65c9a9e363a0a51eaf8873b3dc2ffab45787c5e86bacbf2a6bbca5106828eec95cb2ea534f
a2e64d672a2c69e21589f84daa54a164db28ade473e8009972279cd89c5afaf1b312914256dac66
6e7f824db23f33a9867616898686a1fe63c5
```

Σας δίνεται ότι το ιδιωτικό κλειδί d είναι αρκούντως μικρό, ώστε να επιτρέπεται επίθεση μικρού ιδιωτικού εκθέτη. Να κατασκευάσετε έναν αλγόριθμο και το αντίστοιχο κομψό και αποδοτικό πρόγραμμα σε γλώσσα προγραμματισμού Python ή C (πιθανόν να σας φανεί χρήσιμη η βιβλιοθήκη χειρισμού πολύ μεγάλων αριθμών GMP, που έχει μεταξύ άλλων και ένα φιλικό interface σε Python) που να σας επιτρέπει να σπάσετε το παραπάνω κρυπτοσύστημα. Ποιο είναι το ιδιωτικό κλειδί d; Ποια είναι η υπολογιστική πολυπλοκότητα του αλγορίθμου σας;

(β) Να κατασκευάσετε ένα κομψό και αποδοτικό πρόγραμμα σε γλώσσα Python ή C που να σας επιτρέψει να παραγοντοποιήσετε το παραπάνω n. Ποιοι είναι οι πρώτοι του παράγοντες, p και q;

☠: bonus ερώτημα.

Σύντομες οδηγίες: (α) προσπαθήστε μόνοι σας, (β) συζητήστε με συμφοιτητές σας, (γ) αναζητήστε ιδέες στο διαδίκτυο, με αυτή τη σειρά και αφού αφιερώσετε αρκετό χρόνο σε κάθε στάδιο! Σε κάθε περίπτωση οι απαντήσεις πρέπει να είναι *αυστηρά ατομικές*.